AUSTRALIAN OS9 NEWSLETTER

EDITOR	Gordon Bentzen	(07) 344-3881
SUB-EDITOR	Bob Devries	(07) 372-7816
TREASURER	Don Berrie	(07) 375-3236
LIBRARIAN	Jean-Pierre Jacquet	(07) 372-4675
SUPPORT	Brisbane OS9 Users Group	

Addresses for Correspondence

Editorial Material:

Gordon Bentzen
8 Odin Street
SUNNYBANK Qld 4109

Subscriptions & Library Requests:

Jean-Pierre Jacquet
27 Hampton Street
DURACK Qld 4077

Volume 4 September 1990 Number 8

sides 'No. of cylinders' (in decimal) :Interleave value: (in decimal) @FREE Syntax:
Free [devname] Usage : Displays number of free sectors on a device @GFX
Syntax: RUN GF ce package for
BASIC09 to do c X2 Syntax: RUN
GFX2([path]<funct for BASIC09 to
handle enhanced V Syntax: none
Usage : Graphics handle graphics/
windowing comm ge : Give on-line
help to users will ist of help topics
@IDENT Syntax: ader information
from OS-9 memor memory -s = use
single line output - egins at execution
directory @INIZ Syntax: Iniz <devname>{<devname>} Usage : Attach a device
@INKEY Syntax: RUN INKEY([path],strvar) Usage : BASIC09 subroutine to
input a s abort to
the proc nk to a
memory tents of
text files les into
memory a new
directory memory
module d Syntax:
Merge <p output
@MFREE memory
@MODP odule in
memory ngs -c =
compare ne = link
to module C off obyte nbyte = change obyte at off(set) to nbyte V = verify
module M = mask IRQs U = unmask IRQs @MONTYPE Syntax: Montype [opt]
Usage : Set m monitor m =
monochrome m sage : Creates
and links an OS ROCS Syntax:
Procs [e] Usage em Opts : e =
display all pro e : Prints the
current data d s the current
execution direct new filename>
Usage : Gives t Runb <i-code
module> Usage ntax: Setime
[yy/mm/dd/hh:m ock @SETPR
Syntax: Setpr <p ied process to
num @SHELL Syntax: Shell <arglist> Usage : OS-9 command interpreter
@TMODE Syntax: Tmode [.pathname] [params] Usage : Displays or changes
the ope r /p>
[value] nlink
<modna ntax:
Wcreate [opt] or /wx [-s=type] xpos ypos xsiz ysiz fcol bcol [bord] Usage :
Initialize and create windows Opts : -? = display help -z = read command lines
from stdin -s=type = set screen type for a window on a new screen @XMODE

AUSTRALIAN OS9 NEWSLETTER
Newsletter of the National OS9 User Group
Volume 4 Number 8

EDITOR : Gordon Bentzen
SUBEDITOR : Bob Devries
TREASURER : Don Berrie
LIBRARIAN : Jean-Pierre Jacquet
SUPPORT : Brisbane OS9 Level 2 Users Group

Well here we are at the beginning of yet another subscription year. How time has flown since we first agreed to try to resurrect the Newsletter here in Brisbane! For those of you who were with us back then, you will remember that we stated that we would have to have at least twenty subscribers to make the proposition worthwile.

At the time of writing, we have already achieved more than that required minimum, and therefore we will continue to publish the Newsletter for another year. To those who have sent us their $18.00, thank you for your subscription. For those of you who have not yet resubscribed, please do so, as this will be your last Newsletter. That is, unless we hear from you before the despatch date of the October edition.

We have had, over the years, a reasonable number of renewals, and we interpret this to mean that we are providing the type of material that the readers want. At the same time, we have also had some attrition. This year is no exception, and we have had a couple of letters from members who no longer wish to subscribe for various reasons. All of these reasons involve those people moving to a different type of computer and operating system. We sincerely thank those people for the courtesy of writing to us to inform us of that decision.

Our public-domain software library has continued to grow, as has the number of requests for copying that material. This has resulted in such a work load, that we have had to co-opt another helper to be our librarian. We would like to take this opportunity to introduce to you, Jean-Pierre Jacquet, a member of the Brisbane OS9 Users Group, who has agreed to fill that position. This will, in the near future, result in a number of changes to the way that you request software from our public-domain library.

One of the changes that will be made, is that we will have a database file for all of our archives, and we will be able to search that database for particular types of programmes. For example, if you wished to get a number of programmes which were used for manipulating screens and windows, we would be able to pass that request to our database, and present you with a list of all of those archives that deal with that type of programme. We will be providing further information about these changes in the near future.

Other changes that you will notice, involve the front cover of the Newsletter. Unfortunately, in order to keep the cost of the Newsletter to the same amount that it has been for the last two years, we will be unable to produce the traditional red cover. One hundred and eighty dollars for a new red toner cartridge, is beyond our budget! We hope that not too many of our readers are disappointed, but we would prefer to concentrate on the editorial content, rather than cosmetics. You will also notice that my phone number has changed, thanks to the wisdom of Telecom Australia. Please make a note of the new number.

Well, we here in Brisbane, are looking forward to another exciting year in the world of OS9, particularly with the release of the new machines. We plan to keep you informed and as up to date as possible on that topic. We also plan to keep up with our policy of acquisition of public-domain software, and we will continue to provide that software for the usual nominal copying fee. Remember, however, that if you send disks to us, please format them on your own drives first. This guarantees that you will be able to read them when you get them back.

We would really like to see our local Aussie programmers start to produce some quality software, and are currently investigating mechanisms by which we could encourage this. One of the things that we have noticed, is that in order to continue to receive the PD software, we must provide something in return. So come on folks, give it a go. If you don't think that you have the ability to write the stuff yourself, there are a number of talented people who could transform your ideas into reality. We may be able to get you together!

Anyway, thanks for your support in the past, and we look forward to supporting you in the future.

Cheers, Gordon

oooooooooo0000000000oooooooooo

WANTED: PROGRAMME TESTERS

For some time now, I have been looking for an easier way to run my C-Compiler, and automatically include the necessary library calls to allow the linker to include all of the relevant graphics library calls. You may recall, that I wrote a fairly simple Basic09 programme for the Newsletter which accomplished (in its own small way) just that. (See Australian OS9 Newsletter - May 1989)

For those of you who read/subscribe to the US Rainbow, you may have noticed some time ago, an advertisement for a programme called CCENV, by a previously unknown software vendor called Foxware. Great, I thought, and dutifully called the advertised number. No, we can't accept your Credit Card order, cause we don't have those facilities yet, but sent US$52.45, and all will be OK. Fine, I got to the bank, and bought the necessary currency, and sent off my order.

Well that was in July 1988, and as yet, even after a number of (expensive) international phone calls, I have received neither the software, nor my money back. Oh well, caveat emptor! Let the buyer beware.

Anyway, what all of this is leading to is not primarily an attack on some "dubious quality" software (read vapourware really) vendors, but to ask for volunteers to beta test a programme which does all of the above, and more.

Because of all of the above hassles, I decided to start a project to write (in C - of course) my own version of a windowing front end for the Microware C Compiler. In true fashion, what I thought would be a relatively simple task, turned out to be a major exercise, and I found that I needed both assistance, and incentives to complete the project. At this stage, Bob Devries joined the party, and it has become very much a joint effort since then.

If you have ever used some of the MESS-DOG C Compilers, then this front-end will look very familiar. The following is a very basic description of the programme.

The programme runs in a text window which it sets up for itself (so as not to crud-up you normal system setup), but is mouse driven. A bit like the way one particular CoCo OS9 Word Processor works, but a bit quicker, and lot slicker. It reads its own parameters from a file (located in /dd/SYS), and allows you to specify paths to a number of necessary directories for the compiler files. This is particularly important for users with hard disks.

It allows you to specify the directory for temporary files, specify the name and location of your favourite editor/word processor, list the file to the window, examine the errors produces by the latest compilation of your source, and to change many of these individual preferences on the fly.

Best of all, it cleans up after itself, resets your system back to the way it was at the beginning, removes all of its (three) modules, and returns the memory to the system.

Interested? If so, we would like to supply a free beta version to about 10-12 testers for evaluation, error checking, and constructive criticism. The intention is that, if we get sufficient favourable response from our testers, we will (maybe) market the programme. (The source code currently runs to some 30 pages, so you can see the amount of work that has gone into it).

The testers, if any, will be selected on a first in, first served basis, so if you are interested, please write or call either of us at the addresses below, for further details. Testers will, of course, receive both the beta test, and final release of the programme free of charge.

Don Berrie
25 Irwin Terrace
OXLEY Qld 4075
AUSTRALIA

(07) 375-3236 or +617 375 3236

Bob Devries
21 Virgo Street
INALA Qld 4077
AUSTRALIA

(07) 372-7816 or +617 372 7816

ooooooooooOOOOOOOOOOoooooooooo

COMPUTER WIDOW

This is an open letter to the computer "widows".. there are others of us around. The line of communication with our mates seems to be to leave messages on the screen of the computer, where at least he'll READ them. Things like "I can't get to the clothes line through all the long grass" or "Your dinner has been ready for over an hour - do you like it cold?". I'm not trying to offer solutions - I don't have any - but as in any situation it sometimes helps to know that you are not alone.

How many of you heard "We need to get a computer for the children" - and fell for it! I did, and now, should our children wish to use the computer, they have to book in. And have you ever heard of a teenager who had difficulty getting use of the phone? That's normal in our house.

Our home (I thought) was one purchased with an eye to allowing a little extra room for the teenagers (or, more to the point, allowing us a place to get away from them). After three years we find we are losing ground to computers and all things connected with them. Almost every flat surface is covered by computers and associated paraphernalia and our books are disappearing from the bookshelves to be replaced with computer magazines and manuals. Even the sewing machine has lost its home on occasion during computer meetings but I usually manage to leave enough junk scattered all over it to make it decidedly unattractive for that purpose. (I am also single minded in my determination to retain that area.)

I did manage to have the fridge, washing machine and dishwasher fixed though - he bought new ones. Which just goes to show that every cloud has a silver lining.

That's all I have to say, and thanks to Gordon, Don and Bob for allowing me this space in the newsletter - you may now have your supper. Bye!

ooooooooooOOOOOOOOOOoooooooooo

Disk File Fragmentation
By Bob Devries

Disk fragmentation is something very few of us worry about when using OS9. The operating system is sufficiently smart to allocate and re-allocate disk chunks to suit whatever purpose suits it. There are times, however, when it is desirable to know whether a disk is fragmented badly or not.

Fragmentation is caused by constant saving and deleting of files on a disk. When a disk file is saved onto a disk originally, it is placed in contiguous sectors. When that file is deleted, its sectors are marked as free in the sector allocation table, and are then free to be used for other files. When a new file is to be saved, it may be shorter than the previously deleted one, so a hole is left in the disk structure. This hole will likely not be used unless a file of less than that sector length is to be saved. Frequently single sectors or pairs of sectors are left unused on a disk.

Mostly disk fragmentation is of little concern to floppy disk users, as it is easy to re-organise a disk by copying all the files from it onto a new disk by using 'DSAVE' or a similar programme, which copies the files into contiguous sectors once again. Hard disk users, however do not have this option, so they need to know when their hard drive is in need of attention. The C programme presented here will display the contents of the sector allocation table in a quasi graphical format with the letter 'x' signifying a used sector or cluster, and the letter '_' signifying an unused sector.

There is a fairly good programme sold by Burke and Burke in the USA called 'File System Repack' which has among other hard disk utilities, a disk repack programme, which re-organises the disk without using 'DSAVE'. It takes a fairly long time to do, however, and so it is really useful to know whether it is really necessary.

The programme presented here will work on any screen size, i.e. 32, 40, or 80 columns. To find out which screen size is being used, the system call I$GetStt is used with the SS.SCSIZ option. This system call is not in the defs file 'os9.h', so you'll need to add it as discussed in the May '90 issue of the Newsletter.

Regards,
Bob Devries.

```
/* DU.c - Disk Usage. Writes a map of sector allocation table to stdout */

#include <stdio.h>
#include <direct.h>
#include <os9.h>

union sec {   /* Union to convert 3 byte char to long. Could have used l3tol() */
    long numsec;
    char tsec[4];
};

main(argc,argv)
int argc;
char *argv[];
{
    struct ddsect disk;  /* structure to hold disk lsn0 info */
    int i,fp;
    long sectors;
    unsigned size;
    char desc[80];
    union sec totsec;
    char *store;
    char *malloc();

    if((argc < 2)    (argv[1][0] != '/')) { /* correct params passed ? */
        usage();
        exit(0);
    }
    strcpy(desc,argv[1]);

    if(chdir(desc)==EOF) {   /* whoops, something wrong */
        writeln(1,"Can't do this... '",18);
        writeln(1,argv[0],strlen(argv[0]));
        writeln(1," ",1);
        writeln(1,desc,strlen(desc));
        writeln(1,"'\n",2);
        puts("Can't access that device.");
        usage();
        exit(errno);
    }
    i = strlen(desc);
    desc[i]='@';
    desc[i+1]='\0';

    if((fp = open(desc,_READ)) == EOF) { /* open whole device as file */
        puts("Can't access device for reading.");
        usage();
        exit(errno);
    }
    read(fp,&disk,sizeof(disk));  /* read lsn0 */
    size = disk.dd_map;    /* get allocation map size */
    totsec.tsec[0] = 0;
    totsec.tsec[1] = disk.dd_tot[0];  /* get number of sectors on disk */
    totsec.tsec[2] = disk.dd_tot[1];
    totsec.tsec[3] = disk.dd_tot[2];

    sectors = totsec.numsec;

    if((store = malloc(size)) == NULL) { /* grab some memory to store alloc map */
        puts("memory allocation error.");
        exit(errno);
    }
    lseek(fp,2561,0);    /* seek to alloc map */
    read(fp,store,size);   /* and read it into storage */

    close(fp);     /* and release the disk */

    writeln(1,"Sector Allocation Map for ",26);
    writeln(1,argv[1],strlen(argv[1]));
    writeln(1,"\n",1);
    if(sectors > 4096) {   /* more than 4096 sectors, so tell user */
        puts("This may take a while!");
    }
    display(store,size,sectors);  /* go display map */
    free(store);    /* and free up the memory */
}


/* Display routine for DU.c Disk Usage display programme */

display(store,size,secs)
char *store;
unsigned size;
long secs;
{
    int width,i,j,count,bit;
    long pos;
    long fre,used;
    char byte;
    char line[80];
    char disp[80];

    width = scrnsiz(1);    /* get width of stdout */
    width = (width < 80 && width > 39) ?  40 : width; /* kludge to set size to either 32,40 or 80 */
    width = (width < 40) ?  32 : width;
    pos = 0;
    count = 0;
    fre = 0;
    used = 0;
    for(i=0;i<size;i++) {
        byte = *store++;
        bit = 128;
        for(j=0;j<8;j++) {
            disp[count+j] = (byte & bit) ? 'x' : '_'; /* print either x or _ chars */
            if(disp[count+j] == '_')
                fre++;
            bit = bit >> 1;
        }
```

```
count+=8;
switch (width) {
    case 32:
        if((count==16)    ((pos/8+count/8)>=size)) {
            disp[(count-8)+j]='\0';
            ltoh(pos,line);          /* long to hex conv */
            strcat(line," : ");
            if(i+1==size) {
                disp[strlen(disp)-(size*8-secs)] = '\0';
            }
            strcat(line,disp);
            puts(line);
            line[0] = '\0';
            pos+=count;
            count=0;
        }
        break;
    case 40:
        if((count==24)    ((pos/8+count/8)>=size)) {
            disp[(count-8)+j]='\0';
            ltoh(pos,line);          /* long to hex conv */
            strcat(line," : ");
            if(i+1==size) {
                disp[strlen(disp)-(size*8-secs)] = '\0';
            }
            strcat(line,disp);
            puts(line);
            line[0] = '\0';
            pos+=count;
            count=0;
        }
        break;
    case 80:
        if((count==64)    ((pos/8+count/8)>=size)) {
            disp[(count-8)+j]='\0';
            ltoh(pos,line);          /* long to hex conv */
            strcat(line," : ");
            if(i+1==size) {
                disp[strlen(disp)-(size*8-secs)] = '\0';
            }
            strcat(line,disp);
            puts(line);
            line[0] = '\0';
            pos+=count;
            count=0;
        }
        break;
    default:          /* treat as 80 columns */
        if((count==64)    ((pos/8+count/8)>=size)) {
            disp[(count-8)+j]='\0';
            ltoh(pos,line);          /* long to hex conv */
            strcat(line," : ");
            if(i+1==size) {
                disp[strlen(disp)-(size*8-secs)] = '\0';
            }
            strcat(line,disp);
            puts(line);
```

```
                line[0] = '\0';
                pos+=count;
                count=0;
            }
        }
    }
    used = secs - fre;
    ltoa(fre,line);
    writeln(1,"Free disk space = ",18);  /* inform user of free disk space */
    writeln(1,line,strlen(line));
    writeln(1,"\n",1);
    ltoa(used,line);
    writeln(1,"Used = ",7);   /* and also used, in case he cant add up :-) */
    writeln(1,line,strlen(line));
    writeln(1,"\n",1);
}

scrnsiz(path)
int path;
{
    struct registers reg;   /* get screen size in columns using I$GetStt call */
    int wid;
    reg.rg_a = path;
    reg.rg_b = SS_SCSIZ;
    if(_os9(I_GETSTT,&reg)==0)    /* if system call fails, defaults to 80 columns */
        wid = reg.rg_x;
    else
        wid = 80;
    return(wid);
}

usage()
{
    puts("Usage: du /device");
    puts("       where device is any");
    puts("       RBF device descriptor");
    puts("prints sector allocation map\n");
}

#define DECHEX(x)  ((x) <= 9 ? (x) + '0' : (x) - 10 + 'A')  /*  quick define to convert decimal to hex
*/

ltoh(lnum,str)
long lnum;
char *str;
{
    int i;
    char byte;
    for(i=1;i<=(sizeof(long)*2);i++)
        {
        byte = lnum - ((lnum>>4)<<4);
        lnum = lnum>>4;
        str[(sizeof(long)*2)-i] = DECHEX(byte);
    }
    str[i-1] = '\0';
}
```

```
ltoa(n,s)  /* long to ASCII conversion almost straight from K&R */
long n;
char *s;
{
   int i;
   long sign;

   if((sign = n) < 0)
      n = -n;
   i = 0;
   do {
      s[i++] = n % 10 + '0';
   } while ((n /= 10) > 0);
   if (sign < 0)
      s[i++] = '-';
   s[i] = '\0';
   reverse(s);
}

reverse(s)
char *s;
{
   int c, i, j;
   for(i=0,j=strlen(s)-1;i<j;i++,j--) {
      c = s[i];
      s[i] = s[j];
      s[j] = c;
   }
}
```

oooooooooo0000000000oooooooooo

DYNACALC TERM STRING OFFSETS

We have had a number of requests by people asking us how to run particular programmes from remote terminals. One of the more frequently requested programmes has been the spreadsheet programme, dynacalc. Dynacalc is available for a number of OS9 platforms, including the Color Computer. For most systems other than the CoCo, it came with its own terminal installation programme. For those of you who run terminals on their systems, we publish the following material. We are indebted to Paul Good in the USA for providing these data.

Below are my notes on Dynacalc.trm from the Tandy version 1.00.00 of Dynacalc. Be forewarned that it is incomplete and may contain errors; however, I believe that it is reasonably accurate (I did have access to a number of xxx.trm files and the install utility for a non-CoCo system). So, if you want to experiment, get out your favorite tool (qtip, ded, patch, whatever) and change the values of your choice. If you have to be told to only try this on a backup copy of the file, you should NOT be doing this :-) !!! If you find any errors in the list below, or determine what any of the unknown values are, please let me know.

Paul Good INTERNET: os9paul@gkcl.ists.ca UUCP: ists!gkcl.ists.ca!os9paul

DYNACALC.TRM VARIABLES

```
OFFSET  DESCRIPTION
======  ===========================
88-8A   Cursor on string
8B      FF
8C-8E   Cursor off string
```

8F	FF
90-9E	Terminal set-up string
9F	FF
A0-A6	Terminal "kiss-off" string
A7	FF
A8-AA	Turn printer on string
AB	FF
AC-AE	Turn printer off string
AF	FF
B0-B6	Direct cursor addressing string (see notes below)
B7	FF
B8-BC	Erase to end of line string
BD	FF
BE-C2	Erase to end of page string
C3	FF
C4-CA	Turn highlight on string
CB	FF
CC-D2	Turn highlight off string
D3	FF
D4-D8	Destructive backspace string
D9	FF
DA-DC	Non-destructive backspace string
DD	FF
DE-ED	Terminal name string
EE	04
EF-F1	FF FF FF ???
F2-F4	00 00 00 ???
F5-F6	FF FF ???
F7	Log off character
F8	Use uppercase only regardless of TMODE flag (FF = yes, 00 = no)
F9	Number of line feeds after each line (1-8)
FA	00 ???
FB	Keep Help (00 = yes, FF = no)
FC	Print Borders (00 = yes, FF = no)
FD	Printer width minus one
FE	Pagination (00 = yes, FF = no)
FF	Number of lines per page minus one
100	Up arrow character
101	Down arrow character
102	Left arrow character
103	Right arrow character
104	Home character
105	Jump window character
106	Bell character
107	Get address character
108	Flush type-ahead buffer character
109	Backspace character
10A	Direct cursor addressing row offset
10B	Direct cursor addressing column offset
10C	??? (values of 01, 02, and 03 have been found)
10D	Number of lines on screen
10E	Number of characters per line minus one
10F	Edit overlay character
110	??? (values of 02, and 03 have been found)
111	Edit from entry level character
0680-	Printer device pathlist (limit 60 characters); usually /p

NOTES:

1. Direct cursor addressing - use 88 for column and 99 for row.
   Examples:

ESC Y c r	1B 59 88 99
ESC Y r c	1B 59 99 88

   Don't forget to enter the row and column offset values at 10A-10B.

2. FF in the description area are hex values and appear to be string terminators.

3. ??? indicates that I do not know what the purpose is.

4. All unused bytes in strings are filled with hex FF.

5. Features that are not available on your terminal should be filled with hex FF.

6. Most of the printer variables are for the default value and may be changed from within Dynacalc with the \AP command.

oooooooooo0000000000oooooooooo

Modifications to printer drivers
By Bob Devries

Since I purchased a 4-in-1 card to go with my CRC Super Controller 2, I have been looking at the many programmes which are hard-coded for /P, the serial printer, to see if I could change them to work with /P1, which is the device for the parallel printer. Well, I can say that I have had some success. I thought I would share the information with our newsletter readers who also have parallel printers connected to their Colour Computers.

Here's the information for Home Publisher:

To modify the prn.dmpibmin, which works for DMP130, DMP132, DMP133, here's the modification locations:

Location	old	new
003D	73	2F
003E	2F	50
003F	D0	B1
0094	A9	A8

For the prn.dmp430n and prn.dmp430w:

003C	73	2F
003D	2F	50
003E	D0	B1
0088	B4	B3

For the prn.dmp2100w and prn.dmp2200:

003D	73	2F
003E	2F	50
003F	D0	B1
0089	B4	B3

For the prn.dmp105n:

003C	73	2F
003D	2F	50
003E	D0	B1
008A	B2	B1

For the prn.dmpibm

003D	73	2F
003E	2F	50
003F	D0	B1
0094	A9	A8

For PhantomGraph, the operation is somewhat different, as the printer device name is not contained in data, but in actual programme code. Here's the patches:

For the dmpibm.drv:

00E5	7C	7B
00ED	86	CC
00EF	C6	50
00F0	50	ED
00F1	ED	E4
00F2	E4	CC
00F3	86	31
00F5	E7	ED

Don't forget to verify these modules if you use MODPATCH use the v command at the end of the patch file. If you patch with something else, use the VERIFY command with the u option.

Regards, Bob Devries.

ooooooooooOOOOOOOOOOoooooooooo